

北京圆志科信*射频读写器*

应　用　手　册



RW202XX

地址:北京市通州区通胡大街78号京贸中心大厦1004D
电话：010-64389905　　010-89524306
Web：http://www.yzrfid.com
E-Mail：sales@yzrfid.com

目 录

0.1 更改历史记录

版本	描述	日期
V1.0	PDF 版本第一版发布	2008.4.10
V1.1	为方便客户，增加读卡器发出命令和返回结果举例	2008.8.1
V1.2	增加动态库函数： 向读写卡器下载 Mifare One 卡密钥 用已下载的密钥验证 MifareOne 卡密钥 同时公司地址发生变更	2008.10.9
V1.3	将漏掉的设置设备地址命令补上	2008.12.11

V1.4	rf_init_com();
	rf_init_device_number();
	rf_get_device_number();
	此三个函数在动态库中增加了“设备标识号”
		2009.1.5

V1.5	按产品分类将说明书进行分类: 说明书分别分为 RW202AX.RW202BX.RW202CX.RW202EX,RW202PX	2009.7.30

1. 概述

1.1RW202EX 系列读写器简介：

RW202EX 系列读写模块采用 13.56MHZ 非接触射频技术，内嵌 MFRC500/MFRC632 或其兼容射频基站。用户不必关心射频基站的复杂控制方法，只需通过简单的选定 USB/RS232/RS485 等接口发送命令或操作函数就可以实现对卡片完全的操作。该系列读写模块支持 ISO14443-A Mifare One S50,S70,Ultra Light,MifarePro,FM1208;SAM9600,SAM38400;ISO15693 I CODE SL2 Tagit; ISO14443-B SR176（如果需要读写 SR176 需特殊注明，读卡器硬件需做调整）及其兼容卡片。

1.2 产品型号及之间的区别：

型号	主要区别	对应功能函数
RW202EA	RS232 接口,符合 ISO14443A/B/ISO15693/ISO7816 协议。 支持 Mifare One S50, S70, Ultra Light, MifarePro, FM1208; SAM9600,SAM38400;I CODE SL2 Tagit;SR176（如果需要读写 SR176 需特殊注明，读卡器硬件需做调整）卡片	
RW202EB	RS485 接口,符合 ISO14443A/B/ISO15693/ISO7816 协议。 支持 Mifare One S50, S70, Ultra Light, MifarePro, FM1208; SAM9600,SAM38400;I CODE SL2 Tagit;SR176（如果需要读写 SR176 需特殊注明，读卡器硬件需做调整）卡片	
RW202EC	USB 接口,符合 ISO14443A/B/ISO15693/ISO7816 协议。 支持 Mifare One S50, S70, Ultra Light, MifarePro, FM1208; SAM9600,SAM38400;I CODE SL2 Tagit;SR176（如果需要读写 SR176 需特殊注明，读卡器硬件需做调整）卡片	

1.3 技术指标

- 串口波特率：19200BPS
- 电源： DC5V ± 10%
- 最大功耗： 1.5W
- 环境温度： -10℃ ～ +70℃
- 相对湿度： 35% ～ 95%
- 外形尺寸： 120 * 84 * 25 (mm)
- 重量： 约 100g

1.4 上电状态

读卡器上电后的默认波特率为 19200，绿黄发光二极管闪烁几次后熄灭，读卡器内部红灯长亮:

2. 硬件连接

2.1 RS232 串口的连接

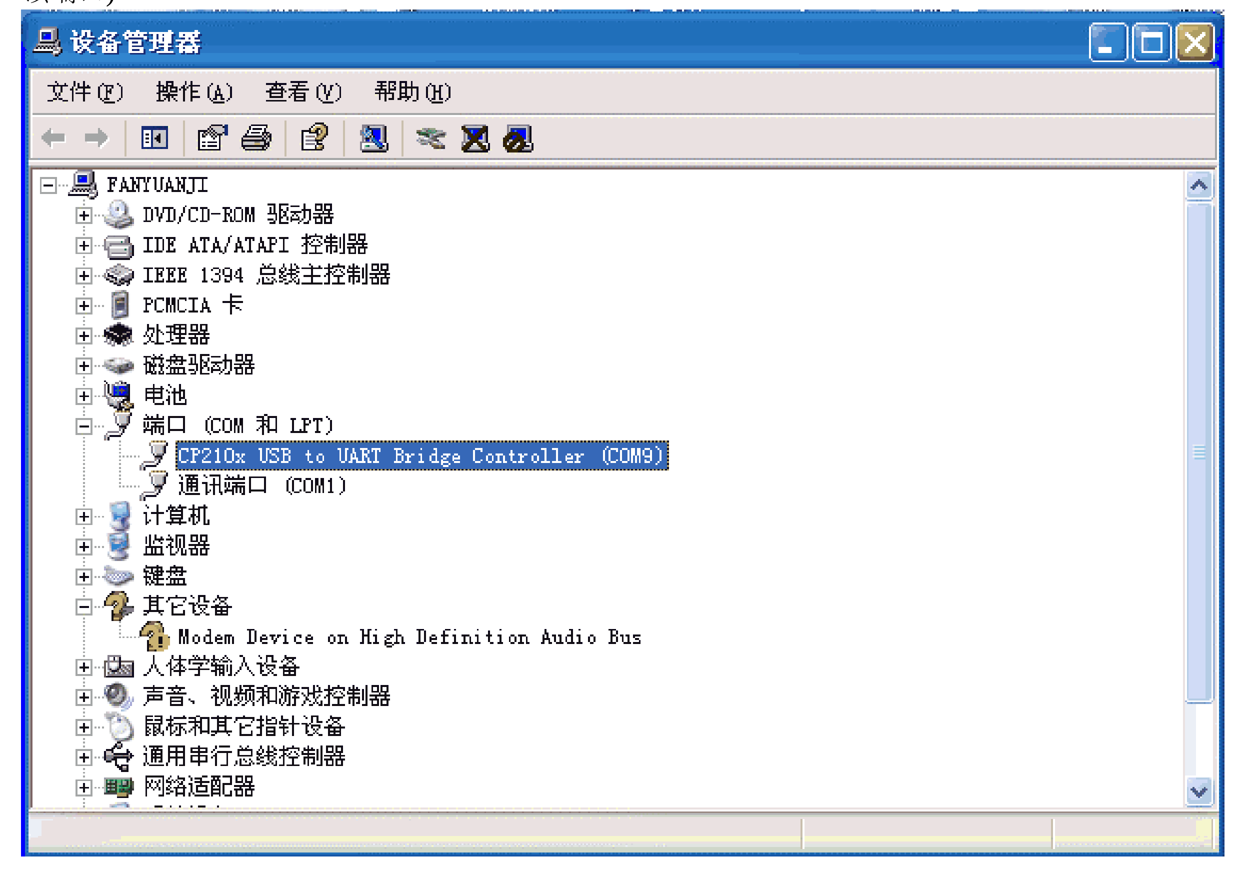
9 针 D 型串口座插到计算机的 COM 口，读卡器接口线上有键盘圆口(取电口)插到计算机键盘口上,将键盘插到线上自带的圆形母头上,这样既不影响键盘的使用,又使读卡器取到+5V 电源;打开 VC++-DEMO 软件选择对应端口及 19200 波特率，点击连接，会出现连接成功或失败提示。

2.2 RS485 串口的连接

读卡器有 4 根出线,白色为 RS485 的 A,灰色为 485 的 B,红色为电源 5V,黑色为地线(电源的负极)，按对应线序进行硬件连线，确保连接无误后，打开 VC++-DEMO 软件选择对应端口及 19200 波特率，点击连接，会出现连接成功或失败提示。

2.3 USB 口的连接

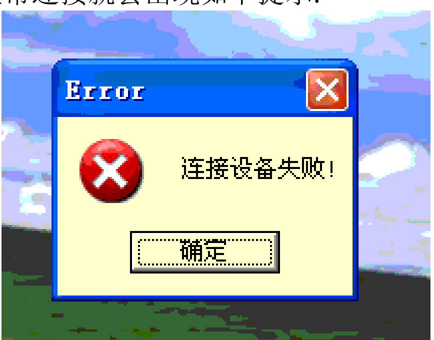
该 USB 可以模拟出 COM 端口;在连接读卡器之前需要先安装驱动程序(CP210x_VCP_Win2K_XP_S2K3.EXE),方法:
双击 **产品光盘\CP2102 驱动程**序\ **CP210x_VCP_Win2K_XP_S2K3.EXE** 安装好驱动程序
插上 USB 读卡器后打开控制面板,双击”系统”,查看系统属性对话框内的”设备管理器”标签,查看端口内即可以显示 CP210X 模拟出的端口号,如图有了一个 COM9(在读卡器软件连接时选用该端口)



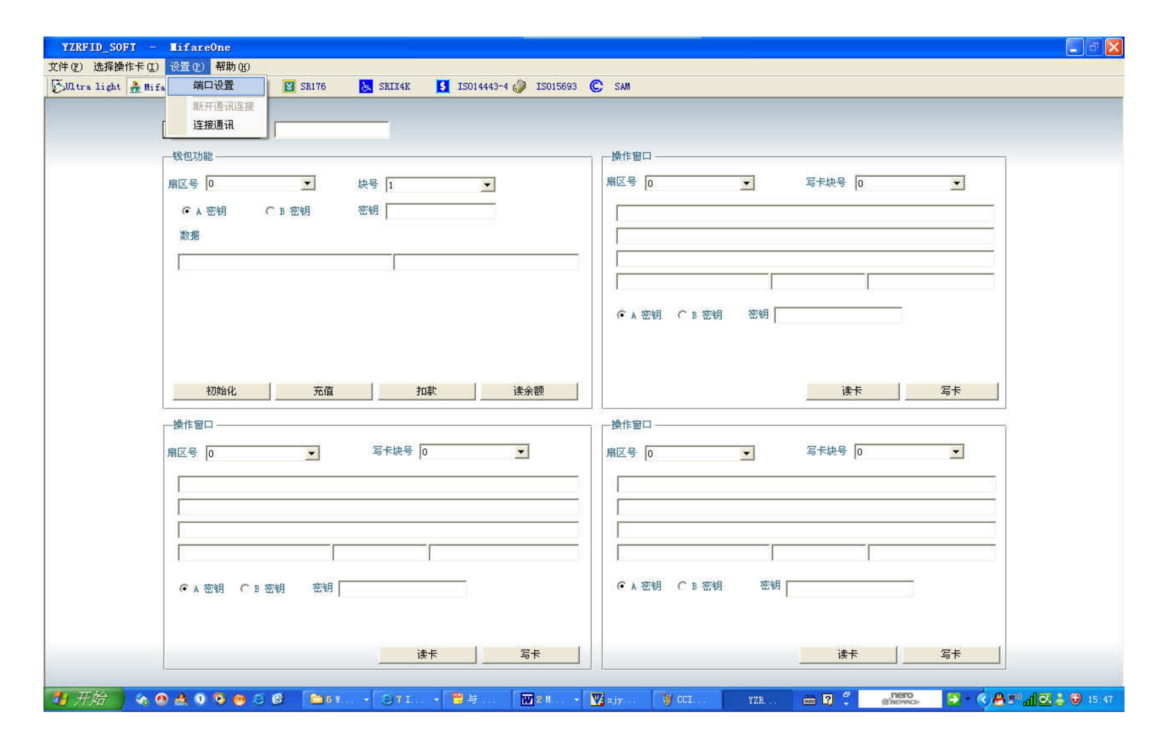
打开软件连接对应端口,比如:COM9 ,19200
打开 VC++-DEMO 软件选择对应端口及 19200 波特率，点击连接，会出现连接成功或失败提示。

注:在 VC++-DEMO 软件启动之前,均需将系统动态库文件 MasterRD.dll　MasterCom.dll 文件拷贝到 windows/system32/目录底下，否则会提示连接设备失败。

打开时如果不能正常连接就会出现如下提示:



此时错误需要去"端口设置"连接端口





此时也可以操作读卡器上的指示灯,蜂鸣器等

M1 读卡测试(新卡初始密码:FFFFFFFFFFFF)

（软件详细介绍请参阅 RW202XXDEMO 软件使用说明）

3．动态链接库说明

3.1 库函数说明

通用函数 功能：获取动态库版本号

原型：int WINAPI lib_ver(unsigned int *nVer)
参数：*nVer：2 字节动态库版本号
返回：成功返回 0

通用函数 功能： DES 算法加密函数

原型：int (WINAPI* des_encrypt)(unsigned char *szOut, unsigned char *szIn , unsigned int inlen, unsigned char *key, unsigned int keylen);
参数：szOut： 输出的 DES 值，长度等于明文长度
szIn： 明文
inlen： 明文长度,8 字节的整数倍
key： 密钥
keylen： 密钥长度,如果大于 8 字节，是 3des,如果小于等于 8 字节单 des.不足补零
返回：成功返回 0

通用函数 功能：DES 算法解密算法函数

原型：int (WINAPI* des_decrypt)(unsigned char *szOut, unsigned char *szIn , unsigned int inlen, unsigned char *key, unsigned int keylen);
参数：szOut： 输出的 DES 值，长度等于密文长度
szIn： 密文
inlen： 密文长度,8 字节的整数倍
key： 密钥
keylen： 密钥长度,如果大于 8 字节，是 3des,如果小于等于 8 字节单 des.不足补零

返回：成功返回 0

通用函数 功能：初始化串口

原型：int WINAPI rf_init_com (unsigned short icdev,int port,long baud)
参数：**icdev：通讯设备标识符，0-65536（新版本的动态库才有此参数）**
port：串口号，取值为 1～4
baud：为通讯波特率 4800～115200
返回：成功返回 0
在进行动态库声明时需要声明“MasterRDnew.dll”动态库

通用函数 功能：指定设备标识

原型：int WINAPI rf_init_device_number (unsigned short icdev,unsigned short icdev)
参数：**icdev：通讯设备标识符，0-65536 （新版本的动态库才有此参数）**
icdev1：要更改的设备标识符号，0-65536
返回：成功返回 0
在进行动态库声明时需要声明“MasterRDnew.dll”动态库

通用函数 功能：读取设备标识

原型：int WINAPI rf_get_device_number (unsigned short icdev,unsigned short *Icdev1)
参数：**icdev：通讯设备标识符，0-65536 （新版本的动态库才有此参数）**
icdev1：存放返回通讯设备标识符
返回：成功返回 0
在进行动态库声明时需要声明“MasterRDnew.dll”动态库

通用函数 功能：读取读写卡器硬件版本号

原型：int WINAPI rf_get_model (unsigned short icdev, unsigned short *Version)
参数：icdev： 通讯设备标识符，0-65536
Version： 存放返回版本信息
返回：成功返回 0

通用函数 功能：读取读写卡器产品序列号

原型：int WINAPI rf_get_snr (unsigned short icdev, unsigned char *Snr)
参数：icdev： 通讯设备标识符，0-65536
Snr： 存放返回读写卡器产品序列号
返回：成功返回 0

通用函数 功能：蜂鸣器控制

原型：int WINAPI rf_beep (unsigned short icdev, unsigned char msec)
参数：icdev： 通讯设备标识符，0-65536
msec： 蜂鸣时限，单位是 10 毫秒
返回：成功返回 0

通用函数 功能：设置指示灯

原型：int WINAPI rf_light(unsigned short icdev, unsigned char color)
参数：icdev： 通讯设备标识符，0-65536
color： 0 ＝ 熄灭黄灯
1 ＝ 熄灭绿灯

2 ＝ 点亮绿灯
3 ＝ 点亮黄灯

PSAM 卡专用函数 功能：设置读写卡器 SAM 卡通讯波特率

原型：int WINAPI rf_init_sam (unsigned short icdev, unsigned char bound)
参数：icdev： 通讯设备标识符，0-65536
bound： 1 字节波特率选择及字节卡片序号：
bit1,bit0(字节波特率选择)
00：9600；01：38400；
返回：成功返回 0

PSAM 卡专用函数 功能：复位 SAM 卡

原型：int WINAPI rf_sam_rst(unsigned short icdev, unsigned char *pData, unsigned char *pMsgLg)
参数：icdev： 通讯设备标识符，0-65536
pDate： 返回的复位信息内容
pMsgLg： 返回复位信息的长度
返回：成功返回 0

PSAM 卡专用函数 功能：向 SAM 发送 COS 命令

原型：int WINAPI rf_sam_cos(unsigned short icdev, unsigned char *command, unsigned char cmdLen ,unsigned char *pData, unsigned char* Length)
参数：icdev： 通讯设备标识符，0-65536
command： COS 命令
cmdLen： COS 命令长度
pDate： 卡片返回的数据，含 SW1、SW2
pMsgLg： 返回数据长度
返回：成功返回 0

非接触卡类型选择专用 功能：设置读写卡器非接触工作方式

原型：int WINAPI rf_init_type (unsigned short icdev, unsigned char type)
参数：icdev：通讯设备标识符，0-65536
type： 读写卡器工作方式
返回：成功返回 0
说明：type='A'：设置为 TYPE_A 方式
type='B'：设置为 TYPE_B 方式
type='1'： 设置为 ISO15693 方式

非接触卡通用函数 功能：设置读写卡器天线状态

原型：int WINAPI rf_antenna_sta (unsigned short icdev, unsigned char model)
参数：icdev：通讯设备标识符，0-65536
model：天线状态
返回：成功返回 0
说明：model=0：关闭天线
model=1：开启天线

Mifare One /Ultralight 专用：寻 TYPE_A 卡

原型：int WINAPI rf_request (unsigned short icdev, unsigned char model, unsigned short *TagType)

参数：icdev：通讯设备标识符，0-65536
model： 寻卡模式
TagType：返回卡类型值
返回：成功返回 0
说明：mode=0x26：寻未进入休眠状态的卡
mode=0x52：寻所有状态的卡

Mifare One/Ultralight 专用 功能：命令已激活的 TYPE_A 卡进入 HALT 状态

原型：int WINAPI rf_halt(unsigned short icdev)
参数：icdev： 通讯设备标识符，0-65536
返回：成功返回 0

Mifare One /Ultralight 专用 功能：读取 MifareOne 卡一块数据

原型：int WINAPI rf_M1_read (unsigned short icdev, unsigned char block, unsigned char *pData,unsigned char *pLen)
参数：icdev： 通讯设备标识符，0-65536
block： M1 卡绝对块号
pData： 读出数据
pLen: 读出数据的长度
返回：成功返回 0

Ultralight 专用 功能：Ultra light 卡防冲撞并激活

原型：int WINAPI int rf_ul_select (word icdev,
unsigned char *pSnr,
unsigned char *pLen)
参数：icdev： [IN] 通讯设备标识符
pSnr： [OUT]返回卡序列号
pLen： [OUT]返回序列号的长度
返回：成功返回 0

Ultralight 专用 功能：向 ultra light 卡中写入一块数据

原型：int WINAPI int rf_ul_write (word icdev,
unsigned char page,
unsigned char *pData)
参数：icdev: [IN] 通讯设备标识符
page： [IN] ultra light 卡页地址（0～0x0f）
pData： [IN] 写入的数据，4 字节
返回：成功返回 0

Mifare One 卡专用 功能： TYPE_A 卡防冲撞

原型：int WINAPI rf_anticoll(unsigned short icdev, unsigned char bcnt, unsigned char *pSnr, unsigned char* pRLength)
参数：icdev：通讯设备标识符，0-65536
bcnt： 卡序列号字节数，取值 4、7、10，Mifare 卡取值 4
pSnr： 返回的卡序列号
pRLength：卡序列号长度
返回：成功返回 0

Mifare One 卡专用 功能：锁定一张 TYPE_A 卡

原型：int WINAPI rf_select (unsigned short icdev, unsigned char *pSnr, unsigned char srcLen, unsigned char *Size)
参数：icdev：　通讯设备标识符，0-65536
pSnr：　卡序列号
srcLen：　卡序列号长度，MifareOne 卡该值等于 4
Size：　返回卡容量
返回：成功返回 0

Mifare One 卡专用 功能：向读写卡器下载 Mifare One 卡密钥

原型：int WINAPI rf_download_key(WORD icdev, unsigned char mode, unsigned char *key);
参数：icdev：通讯设备标识符;
Mode：密钥编号（0--15）
key：密钥，6 字节
返回：成功则返回 0
说明：每 6 个字节为 1 个密钥，0～15 扇区顺序排列
在进行动态库声明时需要声明“MasterRDnew.dll”动态库

Mifare One 卡专用 功能：验证 MifareOne 卡密钥

原型：int WINAPI rf_M1_authentication2(unsigned short icdev, unsigned char model, unsigned char block, unsigned char *key)
参数：icdev：　通讯设备标识符，0-65536
model：　密码验证模式
block：　要验证密码的绝对块号
key：　密钥内容，6 字节
返回：成功返回 0
说明：model=0x60：验证 A 密钥
model=0x61：验证 B 密钥

Mifare One 卡专用 功能：用已下载的密钥验证 MifareOne 卡密钥

原型：int WINAPI rf_M1_authentication1(WORD icdev, unsigned char mode, unsigned char secnr)
参数：icdev：　通讯设备标识符，0-65536
mode：　密码验证模式
secnr：　要验证密码的扇区号（0-15）
返回：成功返回 0
说明：mode=”60”：验证 A 密钥
mode=”61”：验证 B 密钥
在进行动态库声明时需要声明“MasterRDnew.dll”动态库

Mifare One 卡专用 功能：读取 MifareOne 卡一块数据

原型：int WINAPI rf_M1_read (unsigned short icdev, unsigned char block, unsigned char *pData,unsigned char *pLen)
参数：icdev：　通讯设备标识符，0-65536
block：　M1 卡绝对块号
pData：　读出数据
pLen: 读出数据的长度
返回：成功返回 0

Mifare One 卡专用 功能：写入 MifareOne 卡一块数据

原型：int WINAPI rf_M1_write (unsigned short icdev, unsigned char block, unsigned char *data)
参数：icdev：　通讯设备标识符，0-65536
　　　block：　M1 卡绝对块号
　　　data：　写入的数据，16 字节
返回：成功返回 0

Mifare One 卡专用 功能：将 Mifare One 卡某一扇区初始化为钱包

原型：int WINAPI rf_M1_initval (unsigned short icdev, unsigned char block, long value)
参数：icdev：　通讯设备标识符，0-65536
　　　block：　M1 卡绝对块号
　　　value：　初始值，16 进制，低字节在前
返回：成功返回 0

Mifare One 卡专用 功能：读取 Mifare One 卡钱包值

原型：int WINAPI rf_M1_readval(WORD icdev, unsigned char block, long* pValue)
参数：icdev：　通讯设备标识符，0-65536
　　　block：　M1 卡绝对块号
　　　pValue：　返回的值，16 进制，低字节在前
返回：成功返回 0

Mifare One 卡专用 功能： Mifare One 卡扣款

原型：int WINAPI rf_M1_decrement (unsigned short icdev, unsigned char block, long value)
参数：icdev：　通讯设备标识符，0-65536
　　　block：　M1 卡绝对块号
　　　value：　要扣的值，16 进制，低字节在前
返回：成功返回 0

Mifare One 卡专用 功能： Mifare One 卡充值

原型：int WINAPI rf_M1_increment (unsigned short icdev, unsigned char block, long value)
参数：icdev：　通讯设备标识符，0-65536
　　　block：　M1 卡绝对块号
　　　value：　要增加的值，16 进制，低字节在前
返回：成功返回 0

Mifare One 卡专用 功能： Mifare One 卡数据回传

原型：int WINAPI rf_M1_restore (unsigned short icdev, unsigned char block)
参数：icdev：　通讯设备标识符，0-65536
　　　block：　M1 卡绝对块号
返回：成功返回 0
说明：用此函数将指定的块内容传入卡的 buffer，然后可用 rf_M1transfer()函数将 buffer 中数据再传送到另一块中去

Mifare One 卡专用 功能：Mifare One 卡数据传送

原型：int WINAPI rf_M1_transfer (unsigned short icdev, unsigned char block)
参数：icdev：　通讯设备标识符，0-65536

block：　M1 卡绝对块号
返回：成功返回 0
说明：该函数仅在 increment、decrement 和 restore 命令之后调用

ISO14443 TYPE A CPU 卡专用 功能：寻感应区内符合 ISO14443 TYPE_A 标准的 CPU 卡并复位

原型：
```
int WINAPI rf_typea_rst(word icdev,
                        unsigned char  model,
                        unsigned char  *pData,
                        unsigned char  *pMsgLg)
```
参数：icdev：　[IN] 通讯设备标识符
model：　[IN] 寻卡方式
pDate：　[OUT]返回的数据
pMsgLg：　[OUT]返回数据的长度
返回：成功返回 0
说明：mode = 0x26：寻未进入休眠状态的卡
mode = 0x52：寻所有状态的卡
pDate：　4 字节 CSN + 复位信息内容

ISO14443 TYPE A CPU 卡专用 功能：向符合 ISO14443-4 标准的非接触 CPU 卡发送 COS 命令

原型：
```
int WINAPI rf_cos_command(word icdev,
                          unsigned char  *pCommand,
                          unsigned char  cmdLen,
                          unsigned char  *pData,
                          unsigned char  *pMsgLg)
```
功能：向符合 ISO14443-4 标准的 CPU 卡发送 COS 命令
参数：icdev：　[IN] 通讯设备标识符
pCommand：[IN] cos 命令
cmdLen：　[IN] cos 命令长度
pDate：　[OUT]卡片返回的数据，含 SW1、SW2
pMsgLg：　[OUT]返回数据长度
返回：成功则返回 0

ISO14443 TYPE B CPU 卡专用 功能：寻感应区内符合 ISO14443 TYPE_B 标准的卡并激活

原型：int WINAPI rf_atqb (unsigned short icdev, unsigned char model, unsigned char *pData,unsigned char *pMsgLg)
参数：icdev：　通讯设备标识符，0-65536
model:：　寻卡方式
pDate：　返回的复位信息内容
pMsgLg：　返回复位信息长度
返回：成功返回 0
说明：mode= 0：REQB
mode=1：WUPB

ISO14443 TYPE B CPU 卡专用 功能：向符合 ISO14443B 标准的 CPU 卡发送 COS 命令

原型：int WINAPI rf_cos_command (unsigned short icdev, unsigned char *command, unsigned char cmdLen, unsigned char *pData, unsigned char* pMsgLg)
参数：icdev：　通讯设备标识符，0-65536
command：COS 命令

cmdLen：　COS 命令长度
pDate：　卡片返回的数据，含 SW1、SW2
pMsgLg：　返回数据长度
返回：成功返回 0

ISO14443 TYPE B CPU 卡专用 功能：命令已激活的 TYPE_B 卡进入 HALT 状态

原型：int WINAPI rf_hltb (unsigned short icdev, unsigned long PUPI)
参数：icdev：　通讯设备标识符，0-65536
　　　PUPI：　卡片唯一标识符
返回：成功返回 0

SR176 卡专用 功能：：SR176 卡块锁定

原型：int WINAPI int rf_sr176_protectblock(word icdev, unsigned char lockreg)
参数：icdev：　[IN] 通讯设备标识符
　　　lockreg: [IN] LOCKREG
返回：成功则返回 0
说明：SR176 有 16 个块，lockreg 每 1 位控制 2 个块

lockreg	BLOCK	bit_setting	
b7	14 & 15	0:Write Enable	1:Block set as ROM
b6	12 & 13	0:Write Enable	1:Block set as ROM
b5	10 & 11	0:Write Enable	1:Block set as ROM
b4	8 & 9	0:Write Enable	1:Block set as ROM
b3	6 & 7	0:Write Enable	1:Block set as ROM
b2	4 & 5	0:Write Enable	1:Block set as ROM
b1	2 & 3	0:Write Enable	1:Block set as ROM
b0	0 & 1	0:Write Enable	1:Block set as ROM

SR176 卡专用 功能：读 SR176 卡 1 块数据

原型：int WINAPI int rf_sr176_readblock(word icdev,
　　　unsigned char block,
　　　unsigned char *pData,
　　　unsigned char *pLen)
参数：icdev：[IN] 通讯设备标识符
　　　block：[IN] 块地址
　　　pData：[OUT]读出数据
　　　pLen：[OUT]读出数据的长度
返回：成功则返回 0

SR176 卡专用 功能：写 SR176 卡 1 块数据

原型：int WINAPI int rf_sr176_writeblock(word icdev,
　　　unsigned char block,
　　　unsigned char *pData)
参数：icdev：[IN] 通讯设备标识符
　　　block：[IN] 块地址
　　　pData：[IN] 要写入的数据，2 字节
返回：成功则返回 0

SR176 卡专用 功能：命令 ST 卡进入 DESACTIVED 状态

原型：int WINAPI rf_st_completion(word icdev)
参数：icdev： [IN] 通讯设备标识符
返回：成功则返回 0

SR176 卡专用 功能：命令 ST 卡进入 DESACTIVED 状态

原型：int WINAPI rf_st_completion(word icdev)
参数：icdev： [IN] 通讯设备标识符
返回：成功则返回 0

ISO15693 专用 功能：ISO15693_Get_Block_Security

```
原型：int WINAPI ISO15693_Get_Block_Security(word icdev,
                                             unsigned char  model,
                                             unsigned char  *pUID,
                                             unsigned char  block,
                                             unsigned char  number,
                                             unsigned char  *pData,
                                             unsigned char  *pLen)
```

参数：icdev： [IN] 通讯设备标识符
　　　model: [IN] bit0=Select_flag,bit1=Addres_flag,bit2=Option_flag
　　　pUID: [IN] UID 8 字节
　　　block: [IN] 块号
　　　number: [IN] 要读取的块数，< 0x40
　　　pData: [OUT]返回的数据
　　　pLen: [OUT]返回数据的长度
返回：成功则返回 0
说明：Select_flag = 1，只有处于 SELECT 状态的卡执行该命令
　　　Addres_flag = 1，只有 UID 符合的卡执行该命令
　　　设 Option_flag = 0

ISO15693 专用 功能：ISO15693_Get_System_Information

```
原型：int WINAPI ISO15693_Get_System_Information(word icdev,
                                                 unsigned char  model,
                                                 unsigned char  *pUID,
                                                 unsigned char  *pData,
                                                 unsigned char  *pLen)
```

参数：icdev： [IN] 通讯设备标识符
　　　model: [IN] bit0=Select_flag,bit1=Addres_flag,bit2=Option_flag
　　　pUID: [IN] UID 8 字节
　　　pData: [OUT]返回的数据
　　　pLen: [OUT]返回数据的长度
返回：成功则返回 0
说明：model:
　　　bit0 Select_flag = 1，只有处于 SELECT 状态的卡执行该命令
　　　bit1 Addres_flag = 1，只有 UID 符合的卡执行该命令
　　　bit2 设 Option_flag = 0

ISO15693 专用　功能：ISO15693_Inventory(单张卡)

原型：int WINAPI ISO15693_Inventory(word icdev,
unsigned char *pData,
unsigned char *pLen)

参数：icdev：　[IN] 通讯设备标识符
pData:　[OUT]返回的数据，1 字节 DSFID+8 字节 UID
pLen:　[OUT]Pdata 长度

返回：成功则返回 0

ISO15693 专用　功能：ISO15693_Inventorys(多张卡)

原型：int WINAPI ISO15693_Inventorys(word icdev,
unsigned char　*pData,
unsigned char　*pLen)

参数：icdev：　[IN] 通讯设备标识符
pData:　[OUT]返回的数据，每 9 个字节为一组，每组结构为：1 字节 DSFID + 8 字节 UID
pLen:　[OUT]返回数据的长度

返回：成功则返回 0

ISO15693 专用　功能：ISO15693_Lock_AFI

原型：int WINAPI ISO15693_Lock_AFI(word icdev,
unsigned char model,
unsigned char *pUID)

参数：icdev：　[IN] 通讯设备标识符
model:　[IN] bit0=Select_flag,bit1=Addres_flag,bit2=Option_flag
pUID:　[IN] UID 8 字节

返回：成功则返回 0

说明：model:
bit0: Select_flag = 1，只有处于 SELECT 状态的卡执行该命令
bit1: Addres_flag = 1，只有 UID 符合的卡执行该命令
bit2: 如操作的是 TI 卡片设 Option_flag = 1，
如操作的是 I.CODE SLI 卡片设 Option_flag = 0

ISO15693 专用　功能：ISO15693_Lock_Block

原型：int WINAPI ISO15693_Lock_Block(word icdev,
unsigned char　model,
unsigned char　*pUID,
unsigned char　block)

参数：icdev：　[IN] 通讯设备标识符
model:　[IN] bit0=Select_flag,bit1=Addres_flag,bit2=Option_flag
pUID:　[IN] UID 8 字节
block:　[IN] 块号

返回：成功则返回 0

说明：model:
bit0: Select_flag = 1，只有处于 SELECT 状态的卡执行该命令
bit1: Addres_flag = 1，只有 UID 符合的卡执行该命令
bit2: 如操作的是 TI 卡片设 Option_flag = 1，
如操作的是 I.CODE SLI 卡片设 Option_flag = 0

ISO15693 专用 功能：ISO15693_Lock_DSFID

原型：int WINAPI ISO15693_Lock_DSFID(word icdev,
　　　　unsigned char model,
　　　　unsigned char *pUID)
参数：icdev：[IN] 通讯设备标识符
　　model：[IN] bit0=Select_flag,bit1=Addres_flag,bit2=Option_flag
　　pUID：[IN] UID 8 字节
返回：成功则返回 0
说明：model:
　　bit0: Select_flag = 1，只有处于 SELECT 状态的卡执行该命令
　　bit1: Addres_flag = 1，只有 UID 符合的卡执行该命令
　　bit2: 如操作的是 TI 卡片设 Option_flag = 1，
　　　　如操作的是 I.CODE SLI 卡片设 Option_flag = 0

ISO15693 专用 功能：ISO15693_Read

原型：int WINAPI ISO15693_Read(word icdev,
　　　　unsigned char model,
　　　　unsigned char *PUID,
　　　　unsigned char block,
　　　　unsigned char number,
　　　　unsigned char *pData,
　　　　unsigned char *pLen);
参数：icdev：[IN] 通讯设备标识符
　　model：[IN] bit0=Select_flag,bit1=Addres_flag,bit2=Option_flag
　　pUID：[IN] UID 8 字节
　　block：[IN] 块号
　　number：[IN] 要读取的块数，< 0x10
　　pData：[OUT]读出的数据
　　pLen：[OUT]读出数据的长度
返回：成功则返回 0
说明：model:
　　bit0: Select_flag = 1，只有处于 SELECT 状态的卡执行该命令
　　bit1: Addres_flag = 1，只有 UID 符合的卡执行该命令
　　bit2: 如操作的是 TI 卡片设 Option_flag = 1，
　　　　如操作的是 I.CODE SLI 卡片设 Option_flag = 0

ISO15693 专用 功能：ISO15693_Reset_To_Ready

原型：int WINAPI ISO15693_Reset_To_Ready(word icdev,
　　　　unsigned char model,
　　　　unsigned char *pUID)
参数：icdev：[IN] 通讯设备标识符
　　model：[IN] bit0=Select_flag,bit1=Addres_flag,bit2=Option_flag
　　pUID：[IN] UID 8 字节
返回：成功则返回 0
说明：model:
　　bit0: Select_flag = 1，只有处于 SELECT 状态的卡执行该命令
　　bit1: Addres_flag = 1，只有 UID 符合的卡执行该命令

bit2: 设 Option_flag = 0

ISO15693 专用 功能：ISO15693_Select

原型：int WINAPI ISO15693_Select(word icdev, unsigned char *pUID)
参数：icdev： [IN] 通讯设备标识符
pUID: [IN] UID 8 字节
返回：成功则返回 0

ISO15693 专用 功能：ISO15693_Stay_Quiet

原型：int WINAPI ISO15693_Stay_Quiet(word icdev, unsigned char *pUID)
参数：icdev： [IN] 通讯设备标识符
pUID: [IN] UID 8 字节
返回：成功则返回 0

ISO15693 专用 功能：ISO15693_Write

原型：
```
int WINAPI ISO15693_Write(word icdev,
                         unsigned char  model,
                         unsigned char  *pUID,
                         unsigned char  block,
                         unsigned char  *pData);
```
参数：icdev： [IN] 通讯设备标识符
model: [IN] bit0=Select_flag,bit1=Addres_flag,bit2=Option_flag
pUID: [IN] UID 8 字节
block: [IN] 块号
pData: [IN] 要写入的数据，4 字节
返回：成功则返回 0
说明：model:
bit0: Select_flag = 1，只有处于 SELECT 状态的卡执行该命令
bit1: Addres_flag = 1，只有 UID 符合的卡执行该命令
bit2: 如操作的是 TI 卡片设 Option_flag = 1,
如操作的是 I.CODE SLI 卡片设 Option_flag = 0

ISO15693 专用 功能：ISO15693_Write_AFI

原型：
```
int WINAPI ISO15693_Write_AFI(word icdev,
                             unsigned char  model,
                             unsigned char  *pUID,
                             unsigned char  AFI)
```
参数：icdev： [IN] 通讯设备标识符
model: [IN] bit0=Select_flag,bit1=Addres_flag,bit2=Option_flag
pUID: [IN] pUID 8 字节
AFI: [IN] 要写入的 AFI
返回：成功则返回 0
说明：model:
bit0: Select_flag = 1，只有处于 SELECT 状态的卡执行该命令
bit1: Addres_flag = 1，只有 UID 符合的卡执行该命令
bit2: 如操作的是 TI 卡片设 Option_flag = 1,
如操作的是 I.CODE SLI 卡片设 Option_flag = 0

ISO15693专用 功能：ISO15693_Write_DSFID

```
原型：int WINAPI ISO15693_Write_DSFID(word icdev,
                                       unsigned char  model,
                                       unsigned char  *UID,
                                       unsigned char  DSFID)
参数：icdev:  [IN] 通讯设备标识符
      model:  [IN] bit0=Select_flag,bit1=Addres_flag,bit2=Option_flag
      pUID:   [IN] UID 8字节
      DSFID:  [IN] 要写入的DSFID
返回：成功则返回0
说明：model:
      bit0: Select_flag = 1，只有处于SELECT状态的卡执行该命令
      bit1: Addres_flag = 1，只有UID符合的卡执行该命令
      bit2: 如操作的是TI卡片设Option_flag = 1，
            如操作的是I.CODE SLI卡片设Option_flag = 0
```

4．数据通讯协议：

4.1 UART 协议

- UART 接口一帧的数据格式为 1个起始位，8个数据位，无奇偶校验位，1个停止位。
- 波特率：19200
- **发送数据封包格式：**

数据包帧头02	数据包内容	数据包帧尾03

注:0x02、0x03被使用为起始字符、结束字符，0x10被使用为0x02,0x03的辨识字符。因此在通讯的传输数据之中（起始字符0x02，至结束字符0x03之中）的0x02、0x03、0x10字符之前，皆必须补插入0x10做为数据辨识之用。例如起始字符0x02，至结束字符0x03之中有一原始数据为0x020310，补插入辨识字符之后，将变更为0x100210031010。

数据包内容:

模块地址	长度字	命令字	数据域	校验字

模块地址：对于单独使用的模块来说固定为0x0000;
对网络版模块来说为0x0001~0xFFFE;
0xFFFF为广播。

长度字：指明从长度字到校验字的字节数

命令字：本条命令的含义

数据域：该条命令的内容,此项可以为空

校验字：从模块地址到数据域最后一字节的逐字节累加值（最后一字节）。

- **返回数据封包格式：同发送数据封包格式相同**

数据包内容:

模块地址	长度字	接收到的命令字	执行结果	数据域	校验字

模块地址：对与单独使用的模块来说固定为0x0000;
对网络版模块来说为本身的地址;

长度字：　指明从长度字到数据域最后一字节的字节数

命令字：　本条命令的含义

执行结果：0x00 执行正确

0x01---0xFF 执行错误

数据域：　该条命令的内容,返回执行状态和命令内容

校验字：　从模块地址到数据域最后一字节的逐字节累加值（最后一字节）。

4.2 命令列表

基本命令集

命令名称		命令	数据域及解释
设置模块非接触工作方式	发送	0X3A	1 字节非接触读卡 type 说明：type = 'A'：设置为 TYPE_A 方式； type = 'B'：设置为 TYPE_B 方式； type = 'r'：设置为 AT88RF020 卡方式； type = 's'：设置为 ST 卡方式； type = '1'：设置为 ISO15693 方式
	正确返回	0X3A	空
	错误返回	0X3A	
ISO14443-3 TYPE_A 寻卡(mifare one/ultralight)	发送	0X46	1 字节寻卡 model model=0x26 为寻未进入休眠状态的卡； model=0x52 寻所有状态的卡；
	正确返回	0X46	2 字节 TagType（返回卡类型值） pTagType：0x4400 = ultra_light 0x0400 = Mifare_One(S50) 0x0200 = Mifare_One(S70) 0x4403 = Mifare_DESFire 0x0800 = Mifare_Pro 0x0403 = Mifare_ProX
	错误返回	0X46	
ISO14443-3 TYPE_A 卡防冲撞(mifare one)	发送	0X47	1 字节 bcnt (说明：bcnt=0x04)
	正确返回	0X47	4 字节卡序列号
	错误返回	0X47	
ISO14443-3 TYPE_A 选择一张卡(mifare one)	发送	0X48	4 字节卡序列号
	正确返回	0X48	1 字节卡容量

	错误返回	0X48	
用指定的密钥验证 Mifare One 卡某一块(mifare one)	发送	0X4A	1 字节密钥验证 model+ 1 字节绝对块号+ 6 字节密钥 说明：1 字节密钥验证模式：model=0x60 为验证 A 密钥, model=0x61 为验证 B 密钥
	正确返回	0X4A	
	错误返回	0X4A	
读取 Mifare One 卡一块数据(mifare one/ultralight)	发送	0X4B	1 字节绝对块号 说明:S50 块号（0～63）; S70 块号（0～255; 此命令也适合 Ultralight
	正确返回	0X4B	16 字节读出的数据（对于 Ultralight 则读出的数据为连续的 4 个页数据）
	错误返回	0X4B	
写入 Mifare One 卡一块数据(mifare one)	发送	0X4C	1 字节绝对块号 ＋16 字节要写入的数据 说明:S50 块号（0～63）; S70 块号（0～255;）
	正确返回	0X4C	
	错误返回	0X4C	
将 Mifare One 卡某一块初始化为钱包(mifare one)	发送	0X4D	1 字节绝对块号+ 4 字节 16 进制初始金额 说明:S50 块号（0～63）; S70 块号（0～255） ＋4 字节钱包值（低字节在前）
	正确返回	0X4D	
	错误返回	0X4D	
读 Mifare One 钱包值(mifare one)	发送	0X4E	1 字节绝对块号 说明:S50 块号（0～63）; S70 块号（0～255;）
	正确返回	0X4E	4 字节 16 进制金额返回值，低字节在前
	错误返回	0X4E	
Mifare One 钱包充值(mifare one)	发送	0X50	1 字节绝对块号＋4 字节 16 进制金额增加值（低字节在前）
	正确返回	0X50	空
	错误返回	0X50	
Mifare One 钱包扣款(mifare one)	发送	0X4F	1 字节绝对块号＋4 字节 16 进制要扣的金额值（低字节在前）
	正确返回	0X4F	空
	错误返回	0X4F	

将指定块的钱包内容回传至卡的 Buffer (mifare one)	发送	0X51	1 字节绝对块号（说明：用此命令将指定的块数据传入卡的 buffer，然后可用 0X1A 命令将 buffer 中数据，再传送到另一块中去，源地址和目的地址须在同一扇区内；
	正确返回	0X51	空
	错误返回	0X51	
将 Mifare One 卡 Buffer 中的钱包值传送到指定的块中(mifare one)	发送	0X52	1 字节绝对块号（说明：用此命令将 buffer 中数据，再传送到另一块中去，源地址和目的地址须在同一扇区内；
	正确返回	0X52	
	错误返回	0X52	
MIFARE ONE 和 Ultralight 卡休眠 (mifare one/ultralight)	发送	0X29	空
	正确返回	0X29	空
	错误返回	0X29	
Ultra Light 选卡 (Ultralight)	发送	0X33	空
	正确返回	0X33	7 字节卡序列号
	错误返回	0X33	
Ultra Light 读卡 (Ultralight)	发送	0X4B	1 字节起始页号，此命令与 MIFARE ONE 读卡命令相同
	正确返回	0X4B	连续 4 页数据(16 字节)
	错误返回	0X4B	
Ultra Light 写卡 (Ultralight)	发送	0X35	1 字节起始页号 ＋ 4 字节数据
	正确返回	0X35	空
	错误返回	0X35	
TYPE A CPU 卡复位	发送	0X53	1 字节寻卡 model 说明：mode = 0x26：寻未进入休眠状态的卡； mode = 0x52：寻所有状态的卡；
	正确返回	0X53	4 字节 CSN + 复位信息内容
	错误返回	0X53	
TYPE A/TYPE B CPU 卡发送 COS 命令	发送	0X54	COS 命令内容
	正确返回	0X54	卡片返回数据
	错误返回	0X54	

设置 SAM 卡通讯波特率	发送	0X36	**1 字节波特率选择及字节卡片序号 bit1,bit0(字节波特率选择) 00：9600；01：38400；**
	正确返回	0X36	空
	错误返回	0X36	
SAM 卡复位	发送	0X37	空
	正确返回	0X37	复位信息
	错误返回	0X37	
向 SAM 卡发送 COS 命令	发送	0X38	COS 命令内容
	正确返回	0X38	卡片返回数据
	错误返回	0X38	
SR176 选卡		0X60	空
		0X60	返回 1 字节 Chip_ID，返回的 ID 号是随机的，同一张卡每次返回值不一定相同
		0X60	
SR176 取消选定		0X61	空
		0X61	空
		0X61	
SR176 读块		0X62	1 字节块地址
		0X62	读出的 2 字节数据
		0X62	
SR176 写块		0X63	1 字节块地址+2 字节要写入的数据
		0X63	空
		0X63	
SR176 块锁定		0X64	1 字节的 lockreg 数据：说明：SR176 有 16 个块，lockreg 每 1 位控制 2 个块 lockreg BLOCK bit_setting b7 14 & 15 0:Write Enable 1:Block set as ROM b6 12 & 13 0:Write Enable 1:Block set as ROM b5 10 & 11 0:Write Enable 1:Block set as ROM b4 8 & 9 0:Write Enable 1:Block set as ROM b3 6 & 7 0:Write Enable 1:Block set as ROM b2 4 & 5 0:Write Enable 1:Block set as ROM b1 2 & 3 0:Write Enable 1:Block set as ROM b0 0 & 1 0:Write Enable 1:Block set as ROM

		0X64	空
		0X64	
		0x70	空
ISO15693 Inventory		0x70	1 字节 DSFID+8 字节 UID
		0x70	
		0x71	8 字节 UID
ISO15693_Stay_Quiet		0x71	空
		0x71	
		0x72	8 字节 UID
ISO15693 选择卡		0x72	空
		0x72	
		0x73	1 字节 Model+8 字节 UID
ISO15693 复位准备		0x73	空
		0x73	
		0x74	1 字节 Model+8 字节 UID+1 字节块号+1 字节要读取的块数（注：标有*的有此函数字节要读取的块数<10）
ISO15693 读块		0x74	读出的数据
		0x74	
		0x75	1 字节 Model+8 字节 UID+1 字节块号+4 字节要写入的数据
ISO15693 写块		0x75	空
		0x75	
ISO15693 块锁定		0x76	1 字节 Model+8 字节 UID+1 字节块号 说明：model: [IN] bit0=Select_flag, bit1=Addres_flag, bit2=Option_flag Select_flag = 1，只有处于 SELECT 状态的卡执行该命令 Addres_flag = 1，只有 UID 符合的卡执行该命令， 如操作的是 TI 卡片设 Option_flag = 1， 如操作的是 I.CODE SLI 卡片设 Option_flag = 0
		0x76	空
		0x76	

ISO15693 写 AFI		0x77	1 字节 Model+8 字节 UID+1 字节要写入的 AFI 说明：model:　[IN] bit0=Select_flag, bit1=Addres_flag, bit2=Option_flag Select_flag = 1， 只有处于 SELECT 状态的卡执行该命令 Addres_flag = 1，只有 UID 符合的卡执行该命令， 如操作的是 TI 卡片设 Option_flag = 1， 如操作的是 I.CODE SLI 卡片设 Option_flag = 0
		0x77	
		0x77	
15693 锁定 AFI		0x78	1 字节 Model+8 字节 UID 说明：model:　[IN] bit0=Select_flag, bit1=Addres_flag, bit2=Option_flag Select_flag = 1， 只有处于 SELECT 状态的卡执行该命令 Addres_flag = 1，只有 UID 符合的卡执行该命令， 如操作的是 TI 卡片设 Option_flag = 1， 如操作的是 I.CODE SLI 卡片设 Option_flag = 0
		0x78	
		0x78	
ISO15693 写 DSFID		0x79	1 字节 Model+8 字节 UID+1 字节要写入的 DSFID 说明：model:　[IN] bit0=Select_flag, bit1=Addres_flag, bit2=Option_flag Select_flag = 1， 只有处于 SELECT 状态的卡执行该命令 Addres_flag = 1，只有 UID 符合的卡执行该命令， 如操作的是 TI 卡片设 Option_flag = 1， 如操作的是 I.CODE SLI 卡片设 Option_flag = 0
		0x79	
		0x79	
ISO15693 锁 DSFID		0x7A	1 字节 Model+8 字节 UID 说明：model:　[IN] bit0=Select_flag, bit1=Addres_flag, bit2=Option_flag Select_flag = 1， 只有处于 SELECT 状态的卡执行该命令 Addres_flag = 1，只有 UID 符合的卡执行该命令， 如操作的是 TI 卡片设 Option_flag = 1， 如操作的是 I.CODE SLI 卡片设 Option_flag = 0
		0x7A	
		0x7A	

ISO15693 获取系统信息		0x7B	1 字节 Model+8 字节 UID 说明：model:　[IN] bit0=Select_flag, bit1=Addres_flag, bit2=Option_flag Select_flag = 1，只有处于 SELECT 状态的卡执行该命令 Addres_flag = 1，只有 UID 符合的卡执行该命令， 如操作的是 TI 卡片设 Option_flag = 1， 如操作的是 I.CODE SLI 卡片设 Option_flag = 0
		0x7B	返回的数据
		0x7B	
ISO15693 获取块安全信息		0x7C	1 字节 Model+8 字节 UID+1 字节块号 说明：model:　[IN] bit0=Select_flag, bit1=Addres_flag, bit2=Option_flag Select_flag = 1，只有处于 SELECT 状态的卡执行该命令 Addres_flag = 1，只有 UID 符合的卡执行该命令， 如操作的是 TI 卡片设 Option_flag = 1， 如操作的是 I.CODE SLI 卡片设 Option_flag = 0
		0x7C	返回的数据
		0x7C	
设置模块天线状态	发送	0X05	1 字节 Model 说明：Model=0 关闭天线； Model=1 开启天线
	正确返回	0X05	空
	错误返回	0X05	
设置蜂鸣器时间	发送	0X1D	1 字节蜂鸣时限（说明：单位为 10ms)
	正确返回	0X1D	空
	错误返回	0X1D	
初始化串口	发送	0X15	1 字节波特率选择 说明：通讯波特率 0 为 4800；1 为 9600；2 为 14400；3 为 19200；4 为 28800；5 为 38400；6 为 57600；7 为 115200 可选
	正确返回	0X15	
	错误返回	0X15	
控制 LED 灯	发送	0X6A	1 字节 LED 灯的状态控制 说明： 0 ＝ 熄灭黄灯；1 ＝ 熄灭绿灯；2 ＝ 点亮绿灯；3 ＝ 点亮黄灯
	正确返回	0X6A	空
	错误返回	0X6A	
设定设备地址	发送	0X13	2 字节设备地址标识符（地址范围 0-255），比如：0x0101；

	正确返回	0X13	0x00
	错误返回	0X13	

5. 数据通讯协议：

5.1 读卡器通用命令发送接收举例：

端口连接并成功：
【发送数据:】02 00 00 04 15 10 03 1C 03
【接收数据:】02 00 00 10 03 15 00 18 03
控制蜂鸣器声音长短：
【发送数据:】02 00 00 04 1D 50 71 03
【接收数据:】02 00 00 10 03 1D 00 20 03
点亮绿色 LED 灯：
【发送数据:】02 00 00 04 6A 10 02 70 03
【接收数据:】02 00 00 10 03 6A 00 6D 03
熄灭绿色 LED 灯：
【发送数据:】02 00 00 04 6A 01 6F 03
【接收数据:】02 00 00 10 03 6A 00 6D 03
点亮黄色 LED 灯：
【发送数据:】02 00 00 04 6A 10 03 71 03
【接收数据:】02 00 00 10 03 6A 00 6D 03
熄灭黄色 LED 灯：
【发送数据:】02 00 00 04 6A 00 6E 03
【接收数据:】02 00 00 10 03 6A 00 6D 03

5.2 M1 卡发送接收举例：

Mifare S50 卡寻卡并成功
【发送数据:】02 00 00 04 05 00 09 03
【接收数据:】02 00 00 10 03 05 00 08 03
【发送数据:】02 00 00 04 3A 41 7F 03
【接收数据:】02 00 00 10 03 3A 00 3D 03
【发送数据:】02 00 00 04 05 01 0A 03
【接收数据:】02 00 00 10 03 05 00 08 03
【发送数据:】02 00 00 04 46 52 9C 03
【接收数据:】02 00 00 05 46 00 04 00 4F 03
【发送数据:】02 00 00 04 47 04 4F 03
【接收数据:】02 00 00 07 47 00 42 0B C2 08 65 03
【发送数据:】02 00 00 07 48 42 0B C2 08 66 03
【接收数据:】02 00 00 04 48 00 08 54 03
读扇区 0 并成功
【发送数据:】02 00 00 0B 4A 60 00 FF FF FF FF FF FF AF 03
【接收数据:】02 00 00 10 03 4A 00 4D 03
【发送数据:】02 00 00 04 4B 00 4F 03
【接收数据:】02 00 00 13 4B 00 42 0B C2 08 83 08 04 00 62 63 64 65 66 67 68 69 30 03
【发送数据:】02 00 00 04 4B 01 50 03
【接收数据:】02 00 00 13 4B 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 5E 03

【发送数据:】02 00 00 04 4B 10 02 51 03
【接收数据:】02 00 00 13 4B 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 5E 03
【发送数据:】02 00 00 04 4B 10 03 52 03
【接收数据:】02 00 00 13 4B 00 00 00 00 00 00 FF 07 80 69 FF FF FF FF FF FF 47 03

写扇区 0 块 1，将块 1 全写成 1：
【发送数据:】02 00 00 0B 4A 60 01 FF FF FF FF FF FF B0 03
【接收数据:】02 00 00 10 03 4A 00 4D 03
【发送数据:】02 00 00 14 4C 01 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 71 03
【接收数据:】02 00 00 10 03 4C 00 4F 03

将扇区 0 块 1 初始化为钱包，初始值为 100：
【发送数据:】02 00 00 0B 4A 60 01 FF FF FF FF FF FF B0 03
【接收数据:】02 00 00 10 03 4A 00 4D 03
【发送数据:】02 00 00 08 4D 01 64 00 00 00 BA 03
【接收数据:】02 00 00 10 03 4D 00 50 03

将扇区 0 块 1 充值 100：
【发送数据:】02 00 00 0B 4A 60 01 FF FF FF FF FF FF B0 03
【接收数据:】02 00 00 10 03 4A 00 4D 03
【发送数据:】02 00 00 08 50 01 64 00 00 00 BD 03
【接收数据:】02 00 00 10 03 50 00 53 03

将扇区 0 块 1 扣款 50：
【发送数据:】02 00 00 0B 4A 60 01 FF FF FF FF FF FF B0 03
【接收数据:】02 00 00 10 03 4A 00 4D 03
【发送数据:】02 00 00 08 4F 01 32 00 00 00 8A 03
【接收数据:】02 00 00 10 03 4F 00 52 03
读扇区 0 块 1 余额为 150：
【发送数据:】02 00 00 0B 4A 60 01 FF FF FF FF FF FF B0 03
【接收数据:】02 00 00 10 03 4A 00 4D 03
【发送数据:】02 00 00 04 4E 01 53 03
【接收数据:】02 00 00 07 4E 00 96 00 00 00 EB 03
Mifare one 卡休眠
【发送数据:】02 00 00 10 03 29 2C 03
【接收数据:】02 00 00 10 03 29 00 2C 03

5.3 PSAM 卡发送接收举例：

PSAM 卡复位并成功（PSAM9600）：
【发送数据:】02 00 00 04 36 00 3A 03
【接收数据:】02 00 00 10 03 36 00 39 03
【发送数据:】02 00 00 10 03 37 3A 03
【接收数据:】02 00 00 14 37 00 3B 6D 00 00 57 44 29 46 41 86 93 05 6D B0 09 41 56 19 03

SAM 卡发送 COS 指令
【发送数据:】02 00 00 08 38 00 84 00 00 04 C8 03
【接收数据:】02 00 00 09 38 00 D5 74 FA CD 90 00 E1 03

5.4 ISO14443 TYPE A CPU 卡发送接收举例：

FM1208 卡复位并成功：
【发送数据:】02 00 00 04 05 00 09 03
【接收数据:】02 00 00 10 03 05 00 08 03
【发送数据:】02 00 00 04 3A 41 7F 03
【接收数据:】02 00 00 10 03 3A 00 3D 03
【发送数据:】02 00 00 04 05 01 0A 03
【接收数据:】02 00 00 10 03 05 00 08 03
【发送数据:】02 00 00 04 53 52 A9 03
【接收数据:】02 00 00 0F 53 00 16 61 1B 82 10 10 78 80 90 10 02 20 90 00 C0 03
FM1208 发送 COS 指令(0084000004)并成功返回：
【发送数据:】02 00 00 08 54 00 84 00 00 04 E4 03
【接收数据:】02 00 00 09 54 00 7B A3 5F 28 90 00 92 03

5.5 Ultralight 卡发送接收举例：

Ultralight 寻卡
【发送数据:】02 00 00 04 05 00 09 03
【接收数据:】02 00 00 10 03 05 00 08 03
【发送数据:】02 00 00 04 3A 41 7F 03
【接收数据:】02 00 00 10 03 3A 00 3D 03
【发送数据:】02 00 00 04 05 01 0A 03
【接收数据:】02 00 00 10 03 05 00 08 03
【发送数据:】02 00 00 04 46 52 9C 03
【接收数据:】02 00 00 05 46 00 44 00 8F 03
【发送数据:】02 00 00 10 03 33 36 03
【接收数据:】02 00 00 0A 33 00 04 61 7D B1 CA 10 02 80 1C 03

Ultralight 读卡（读出全部的 16 页）
【发送数据:】02 00 00 04 4B 00 4F 03
【接收数据:】02 00 00 13 4B 00 04 61 7D 90 B1 CA 10 02 80 F9 48 00 00 00 00 00 00 0E 03
【发送数据:】02 00 00 04 4B 04 53 03
【接收数据:】02 00 00 13 4B 00 FF FF FF FF 00 00 00 00 00 00 00 00 00 00 00 00 5A 03
【发送数据:】02 00 00 04 4B 08 57 03
【接收数据:】02 00 00 13 4B 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 5E 03
【发送数据:】02 00 00 04 4B 0C 5B 03
【接收数据:】02 00 00 13 4B 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 5E 03

Ultralight 写第 4 页，内容：FFFFFFFF
【发送数据:】02 00 00 08 35 04 FF FF FF FF 3D 03
【接收数据:】02 00 00 10 03 35 00 38 03

5.6 ISO15693 卡 I CODE 2 发送接收举例：

ISO15693 得到系统信息
【发送数据:】02 00 00 04 3A 31 6F 03
【接收数据:】02 00 00 10 03 3A 00 3D 03
【发送数据:】02 00 00 0C 7B 00 F0 F4 12 00 00 00 00 00 7D 03
【接收数据:】02 00 00 11 7B 00 0F 45 84 AD 10 02 00 01 04 E0 00 00 1B 10 03 01 17 03

ISO15693 读块（读出全部的块）

【发送数据:】02 00 00 0E 74 10 02 45 84 AD 10 02 00 01 04 E0 00 0E EF 03
【接收数据:】02 00 00 3B 74 00 11 11 11 11 22 22 22 22 33 33 33 33 44 44 44 44 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 57 03
【发送数据:】02 00 00 0E 74 10 02 45 84 AD 10 02 00 01 04 E0 0E 0E FD 03
【接收数据:】02 00 00 3B 74 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 57 5F 4F 4B FF 03
【发送数据:】02 00 00 0E 7C 10 02 45 84 AD 10 02 00 01 04 E0 00 1C 05 03
【接收数据:】02 00 00 1F 7C 00 01 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 9C 03

ISO15693 写第 5 块，内容：55555555

【发送数据:】02 00 00 11 75 10 02 45 84 AD 10 02 00 01 04 E0 05 55 55 55 55 3E 03
【接收数据:】02 00 00 10 03 75 00 78 03

ISO15693 写 DSFID 内容 00

【发送数据:】02 00 00 0D 79 10 02 45 84 AD 10 02 00 01 04 E0 00 E5 03
【接收数据:】02 00 00 10 03 79 00 7C 03